Revista de Estudios Empresariales. Segunda época.
Número: 2 (2015). Páginas: 24 – 48 DOI: 10.17561/ree.v0i2.2739

PROCESOS DE INNOVACIÓN EN LA PRODUCCIÓN DE *SOFTWARE* EN ARGENTINA. UN ESTUDIO DE CASO

INNOVATION PROCESSES IN THE *SOFTWARE* PRODUCTION FROM ARGENTINA. A CASE STUDY

Alejandro Morero, Hernán (Universidad Nacional de Córdoba (Argentina) *
Borrastero, Carina (Universidad Nacional de Córdoba (Argentina) **
Motta, Jorge José (Universidad Nacional de Córdoba (Argentina) ***

RESUMEN

El objetivo general del trabajo es analizar en profundidad los procesos de innovación y desarrollo de capacidades en una firma de *software* y servicios informáticos (SSI) de Argentina. En función de ello se indaga sobre el proceso de innovación en dicha actividad; el papel de las competencias técnicas y organizacionales en este proceso; las particularidades de la organización del trabajo y su influencia en el desarrollo de capacidades y resultados de innovación; las vinculaciones y su incidencia en la propensión a innovar y capacidades de la firma. Se seleccionó una empresa de *software* dinámica de la ciudad de Córdoba y se estudiaron de modo cualitativo y comparativo dos proyectos tecnológicos innovadores. El marco teórico de referencia está basado centralmente en la nueva literatura sobre Economía del Conocimiento, en particular, en antecedentes de investigación sobre un tipo especial de servicios que en las últimas décadas ha generado un interés creciente: *Knowledge Intensive Business Services* (KIBS), caracterizados por producir insumos inmateriales intensivos en conocimiento para los procesos de negocio en otras organizaciones que dependen fuertemente del conocimiento profesional. Entre ellos, el sector de SSI es uno de los más innovadores en países desarrollados y también en algunas economías emergentes. En este contexto, cobra relevancia un estudio que profundice en la naturaleza misma del proceso innovador en empresas de SSI. Ello puede generar aportes significativos al diseño de instrumentos de medición de la innovación que se encuentran en diversos estadios de desarrollo y prueba.

Palabras claves: Sector de s*oftware*, proceso de innovación, estudios de caso, Argentina.
JEL: O31, L89, L17.

ABSTRACT

The objective of the paper is to analyze the innovation and capacity building process in the *software* production in Argentina, throw a deep unique case study. The dimensions taken into account are the innovation process itself and the role of technical and organizational capacities; the particularities of the work organization of the firm and their relation on the innovation results and the competences building; and the influences of linkages over the firm's innovative propensity. As a case study, a dynamic *software* firm from Córdoba city (Argentina) was selected, and were studied form a qualitative approach two innovative technological projects from the firm. The theoretical roots of the study are in innovation and

* Centro de Investigaciones y Estudios sobre Cultura y Sociedad – CONICET, Rondeau 467, Primer Piso, 5000, Córdoba, Argentina. hernanmorero@eco.uncor.edu
** Centro de Investigaciones y Estudios sobre Cultura y Sociedad – CONICET Rondeau 467, Primer Piso, 5000, Córdoba, Argentina. carinaborrastero@conicet.gov.ar
*** Instituto de Economía y Finanzas – FCE –. jjmotta@eco.unc.edu.ar

Recibido: Junio de 2015. Aceptado: Octubre de 2015.

knowledge economics, and in the Knowledge Intensive Business Services (KIBS) literature. KIBS are characterized by produce intangibles knowledge-intensive which require high levels of qualified staff and usually low levels of capital goods. Within the KIBS sectors, the *software* and related IT services sector is one of the most innovative in developed economies and in some catching-up economies as well. In this background, a study of the nature of the innovation process in the *software* sector is highly relevant, as a contribution to the design of specific innovation surveys to the sector, that are actually in diverse stages of development and testing.

Key Words: Software sector, innovation process, case studies, Argentina. **JEL:** O31, L89, L17.

1. MARCO INTRODUCTORIO

El objetivo general de este trabajo es analizar en profundidad los procesos de innovación y desarrollo de capacidades en una firma de *software* y servicios informáticos (SSI) de Argentina. Los objetivos específicos de la investigación son: i) caracterizar el proceso de innovación en la actividad de producción de *software* a partir del análisis de proyectos tecnológicos innovadores con impacto en el desempeño económico de la empresa; ii) examinar el papel que cumplen las competencias técnicas y organizacionales de la firma de *software* en sus procesos de innovación; iii) estudiar las particularidades que asume la organización del trabajo en los proyectos y su influencia en el desarrollo de capacidades y en los resultados de innovación; y iv) analizar las vinculaciones de esta empresa con el entorno y su incidencia en la propensión a innovar y las capacidades de la firma.

Con este análisis, nos interesa contribuir al estudio de los procesos de innovación y desarrollo de capacidades en empresas de SSI, reconociendo que la definición misma de innovación en este tipo de firmas, así como las estrategias metodológicas de aproximación empírica al problema, forman parte de un debate aún abierto.

En función de los objetivos y horizontes señalados proponemos un abordaje cualitativo, sobre la base de un estudio de caso en profundidad. Seleccionamos una empresa de *software* de la ciudad de Córdoba (Argentina) con perfil innovador, denominada *Machinalis*. Al interior de la firma estudiamos de modo comparativo las particularidades de dos proyectos tecnológicos innovadores.

De manera que, si bien es limitada la posibilidad de generalización de los resultados obtenidos a partir de un estudio de caso, consideramos que el presente trabajo constituye un primer paso en dirección a conocer con mayor profundidad y cobertura los procesos de innovación en empresas de SSI de una economía emergente como la argentina.

Nuestro marco teórico de referencia está basado en aportes de la Teoría Evolucionista, el Institucionalismo, la Economía de la Innovación y, particularmente, la nueva literatura sobre Economía del Conocimiento (Nelson, 1982; Lundvall, 1988; Lundvall, 1992; Lundvall y Johnson, 1994; Nonaka, 1995; Lundvall, 1996; Ancori *et al*., *2000*; Cowan *et al*., 2000; Ernst y Lundvall, 2004).

Desde finales de los años ‘90 distintos autores se ocuparon de investigar en profundidad las características diferenciales de la innovación en los sectores de servicios, en comparación con las industrias manufactureras (Gallouj y Weinstein, 1997). En particular, los estudios en economía de la innovación tienden a resaltar ciertos rasgos propios de la producción de servicios que inciden en la naturaleza misma de los procesos de innovación (Drejer, 2004; Gallouj y Savona, 2009): la inmaterialidad del producto, una continua reconfiguración de la

oferta, la co-producción y simultaneidad de la provisión y el consumo (Gallouj y Savona, 2009).

En las últimas décadas se ha forjado un interés creciente por un tipo particular de servicio: los servicios intensivos en conocimiento o *Knowledge Intensive Business Services* (KIBS). Éstos se caracterizan por producir insumos inmateriales intensivos en conocimiento para los procesos de negocio en otras organizaciones que dependen fuertemente del conocimiento profesional (Miles, 2005; Muller y Doloreux, 2009). Este tipo de sectores son altamente innovadores, por ejemplo: a 2013, los sectores KIBS de la UE mantenían niveles de alrededor del 39 por 100 de su empleo dedicado a actividades de I+D, mientras los sectores manufactureros de mediana y alta tecnología poseen niveles de entre el 5 y el 6 por 100 del personal en este tipo de actividades (Eurostat). Entre los KIBS, el sector de SSI es uno de los más innovadores en países desarrollados y también en algunas economías emergentes (Niosi *et al*., 2012).

Al mismo tiempo, tomando en cuenta que un recurso productivo fundamental de las empresas de base tecnológica es el conocimiento, es preciso considerar tanto las formas en que éste circula en la organización (Nonaka, 1995), sus tipos (know what, know why, know how y know who) y fuentes (Lundvall y Johnson, 1994; Lundvall, 1996; Foray y Lundvall, 1998), así como las diversas modalidades de aprendizaje que se presentan (learning by doing (Arrow, 1962), learning by failing (Maidique y Zirger, 1985), learning by using (Rosenberg, 1982), learning by interacting (Lundvall, 1988), actividades innovadoras en general, de I+D, incorporación de recursos humanos, etc.

La producción de *software* es una actividad compleja que involucra un conjunto de fases altamente creativas y de servicios, en la que las actividades innovadoras forman parte del propio proceso productivo. En su extremo más virtuoso, las conductas creativas de las firmas serán las que habiliten la emergencia de innovaciones, en oposición a las conductas de tipo adaptativo (Schumpeter, 1947; Antonelli, 2008). Ello tanto en términos estrictamente técnicos, como organizacionales y comerciales. En este marco cobra relevancia un estudio que profundice en la naturaleza del proceso innovador en empresas de SSI y en la dinámica de su integración al proceso de producción. Al mismo tiempo, una mejor comprensión de la naturaleza de la innovación puede generar aportes significativos al diseño de instrumentos de medición de la innovación que actualmente se encuentran en diversos estadios de desarrollo y prueba. En función de ello, en este trabajo exponemos los resultados de la investigación cualitativa que permitió indagar *cómo* se da el proceso innovador en los proyectos innovadores de una empresa dinámica que opera en un segmento *top knowledge* del sector de SSI local.

El trabajo está movilizado por una serie de interrogantes en torno a la naturaleza de los procesos de innovación generados en el marco del desarrollo de dichos proyectos: ¿Qué aspectos internos y externos de la firma inciden en la realización de un proyecto innovador en una empresa de *software*, así como en su éxito o fracaso? ¿Qué papel cumplen las capacidades organizacionales y tecnológicas de la empresa en el desarrollo de proyectos innovadores? ¿Qué aprendizajes surgen del desarrollo de un proyecto innovador y cómo inciden en la dinámica de la firma? Y en términos más generales: ¿Qué caracteriza a una empresa innovadora del sector de SSI? ¿Cuál es la dinámica de desarrollo de capacidades en este tipo de firmas? ¿Cuál es el rol de las vinculaciones con otros actores en el proceso innovador de una empresa de SSI? ¿Cuáles son los determinantes de la innovación en PYMES de *software*?

El artículo se organiza del siguiente modo. En la sección 2 se presenta el marco metodológico de la investigación. En la sección 3 se presenta el caso a partir de la descripción

exhaustiva de las características de los proyectos tecnológicos seleccionados, sistematizadas en torno a las variables relevantes surgidas del estudio: orígenes y fases de desarrollo del proyecto; organización del trabajo al interior del proyecto; aprendizajes, capacidades y vinculaciones; medición de la *performance* y resultados de innovación obtenidos. En la sección 4 se realiza un análisis comparativo de los proyectos tecnológicos en torno a las variables seleccionadas. Por último, en la sección 5 se presentan algunas reflexiones finales que pueden orientar futuras líneas de investigación.

2. METODOLOGÍA

Desarrollamos una estrategia metodológica cualitativa. El estudio realizado es de tipo exploratorio y descriptivo, especialmente indicado para conseguir una familiarización con un fenómeno aun escasamente comprendido, para generar nuevas ideas que permitan formular preguntas y nuevas hipótesis, y para conocer la perspectiva de los actores protagonistas de los fenómenos bajo estudio (Eisenhardt, 1989; Denzin y Lincoln, 2005; Neergaard y Ulhøi, 2007; Yin, 2009). Para ello planteamos el estudio de un caso en profundidad con el objeto de indagar acerca del "cómo" se da el proceso innovador en una empresa de producción de SSI local.

En el estudio analizamos un caso único, tomando como unidad de análisis los proyectos tecnológicos de la empresa seleccionada. Según Yin (2009), diseñar un estudio de caso único es pertinente en ocasión de casos críticos (*critical cases*), casos extremos (*extreme cases*), casos típicos (*typical cases*), casos revelatorios (*revelatory cases*) o casos longitudinales. Se justifica la elección de un caso típico cuando el objetivo es *"capturar circunstancias y condiciones de una situación común y cotidiana"* (Yin, 2009) y el caso *"puede representar un 'proyecto' típico dentro de muchos proyectos diferentes (...)"* (Ibídem). Por su parte, un caso revelatorio se manifiesta *"cuando el investigador tiene la oportunidad de observar y analizar un fenómeno previamente inaccesible para la investigación en ciencias sociales"* (Yin, 2009). Según Neergaard (2007) los casos revelatorios pueden ser vistos como una subespecie de los casos típicos.

De manera que se buscó una empresa dinámica del sector de *software* que opere en un segmento *top knowledge* del mercado, de modo que ofrezca nuevas visiones, por un lado, como caso típico respecto a las características del proceso de innovación en firmas del sector de elevada complejidad tecnológica y, por otro lado, como caso revelador de rasgos distintivos de proyectos tecnológicos altamente innovadores. Esta doble estrategia brinda la oportunidad de observar y analizar la naturaleza de la innovación en este tipo de sectores a través del estudio de proyectos como unidad de análisis (en lugar de la empresa, nivel de análisis más usual en la literatura de referencia).

A partir de la realización de entrevistas con informantes clave de la firma y la revisión de documentación organizacional, el análisis se centró en el estudio en profundidad de dos proyectos tecnológicos relevantes para la dinámica innovadora y económica de la empresa. La firma Machinalis se caracteriza por realizar dos tipos de proyectos tecnológicos:

- Proyectos comerciales (*client led projects*): se trata de desarrollos a medida típicamente comerciales, orientados a la satisfacción de las necesidades de un cliente.
- Proyectos internos (*internally led projects*): se trata de desarrollos no orientados a clientes ni con objetivos estrictamente comerciales, de tipo *open source.*

En virtud de esta particularidad de la empresa y con el objeto de obtener una mayor riqueza interpretativa, se seleccionaron dos proyectos tecnológicos representativos de cada uno de los tipos mencionados:

- El proyecto *Proflow*, que consiste en el desarrollo de un *software* de gestión integral para la industria de tratamiento de aeropartes. Se trata de un proyecto de tipo comercial.
- El proyecto *Quepy*, que consiste en el desarrollo de un *software* conversor de preguntas en lenguaje natural a consultas de bases de datos. Se trata de un proyecto interno de la empresa.

Los proyectos fueron estudiados diacrónicamente en función de cuatro ejes de análisis:

1) Su origen y las fases de su desarrollo.
2) Las particularidades que asume la organización del trabajo al interior de cada proyecto.
3) Los aprendizajes surgidos del proyecto y el desarrollo de capacidades y vinculaciones.
4) Las medidas de *performance* y resultados de innovación de los proyectos.

Se realizaron ocho entrevistas semiestructuradas en torno a estos ejes a distintos integrantes de la empresa, con una duración de entre 40 y 90 minutos cada una: Director Ejecutivo (o CEO, por *Chief Executive Officer*), Responsable Coordinador de Proyectos, *Project Managers*, *Technical Leaders* y desarrolladores.

A partir del estudio diacrónico, se analizaron en términos comparativos las características de los proyectos relativas a las variables de análisis mencionadas desde la perspectiva de los procesos de innovación en la firma.

A lo largo de la reconstrucción del caso se mencionará a los integrantes de la empresa por su nombre de pila y la inicial de su apellido sumado a su puesto en el proyecto del que forman parte. Al cliente nos referimos mediante el seudónimo *X* para mencionar a la empresa y *J* para mencionar a su dueño/CEO/jefe de operaciones según corresponda.

3. LOS PROYECTOS INNOVADORES DE MACHINALIS

Machinalis es una empresa de la ciudad de Córdoba, Argentina, especializada en el desarrollo de *software* y servicios en las áreas de Inteligencia Artificial (en particular, *Natural Language Processing*, *Data Mining* y *Machine Learning*), *Data Processing, Big Data* o *Data Science* (incluyendo *Extraction, Data Analysis, Data Modelling* y *Data Visualization*) y Servicios de Desarrollo Complejo (*Complex Web Development*, *Desktop Development* y *Process Automation*).

La firma se dedica principalmente al desarrollo de *software* a medida, tiene una antigüedad de 4 años, exporta el 100% de su producción y cuenta en la actualidad con una plantilla de alrededor de 35 empleados. Entre ellos, la mayoría tiene un alto nivel de cualificación formal, principalmente educación universitaria completa (el 65 por 100 de la planta) y posgrado completo (15 por 100 de los empleados).

A continuación reportamos los resultados obtenidos del relevamiento de los proyectos que conforman la unidad de análisis del presente trabajo.

3.1. Un proyecto para clientes: *Proflow*

Proflow es un sistema de gestión integral desarrollado a medida por Machinalis para una empresa estadounidense de tratamiento de aeropartes. El sistema permite al cliente manejar

toda su información en las distintas fases del proceso productivo (desde los pedidos de presupuesto, la producción, facturación, entrega, etc.). Todos los procesos de negocios están automatizados. Permite a todos los sectores de la empresa monitorizar en qué fase del proceso se encuentra cada orden de trabajo, lo que se traduce principalmente en ahorro de tiempo en cada operación[1].

- *Origen y fases de desarrollo del proyecto*

El proyecto surgió a partir de una persona conocida de un amigo del actual CEO de Machinalis, quien tiene un hermano en los Estados Unidos que era dueño y también CEO (J) de la empresa X. En aquel momento, J pretendía desarrollar un sistema de información a medida que permitiera agilizar los procesos de la planta, dada la inexistencia de un *software* específico para el segmento productivo del cromado y tratamientos químicos de aeropartes[2]. La mencionada persona en común entre los CEOs de X y Machinalis, los puso en contacto como posible vía para satisfacer dicha necesidad. J decidió contactar a Machinalis porque reconocía que escalar el *software* genérico que venía usando hasta el momento resultaría igual de costoso que desarrollar uno nuevo que le daría a la empresa muchas más ventajas. Fue la empresa de J la que realizó el estudio previo y llegó a la conclusión de que no existía un *software* adaptado. Las alternativas para contratar una empresa de *software* que realizara el desarrollo a medida eran Machinalis, otra empresa de Argentina y una de Colombia. J planteó que necesitaba una empresa que elaborara un sistema de gestión totalmente adaptado y que pudiera continuar desarrollando luego según las nuevas necesidades que fueran surgiendo, proveyéndoles a la vez el soporte técnico. Según los entrevistados, la elección de Machinalis surgió de un proceso comercial típico, donde el cliente evaluó que el proyecto presentado por la empresa sería el menos costoso. A partir de este primer contacto, quien sería el primer *Project Manager* de *Proflow* viajó a EEUU para visitar la planta e identificar en qué consistía exactamente lo que el potencial cliente necesitaba. A partir de esa entrevista inicial se comenzaron a producir los primeros requerimientos de *software*, y a partir de un documento elaborado por el *Project Manager* para presentar en Machinalis surgió el proyecto propiamente dicho, se conformó un grupo de trabajo y comenzaron las tareas de desarrollo. Fueron tres las razones principales que impulsaron a Machinalis a definir desarrollar el proyecto: 1) la última etapa programada del proyecto involucraba originalmente el desarrollo de herramientas basadas en los campos de experticia de la empresa y en ese sentido resultaba tecnológicamente desafiante, según destacan los entrevistados; 2) se trataba de un proyecto que generaría ingresos importantes; y 3) permitiría la inserción en el mercado estadounidense, objetivo de la empresa.

La primera fase del proyecto *Proflow* comprendió el desarrollo del sistema de gestión integral en sí, con una duración de dos años. Según los entrevistados, el periodo inicial de esta fase fue fundamental para lograr el objetivo *sine qua non* de conocer la industria de destino, hasta ese momento completamente ajena a la experiencia de Machinalis:

> *No hay forma de construir un software hoy en día sin tener un contacto directo con el cliente. (…) Porque incluso los clientes tampoco saben qué es lo que necesitan, tienen una visión, pero parte de nuestro trabajo es bajar esa visión a tierra, desmenuzarla y*

[1] Por ejemplo: la atención telefónica al cliente antes insumía una media hora y con el *software* las respuestas pueden darse instantáneamente; la planificación de las órdenes de producción tardaba unos cinco días y con el *software* se realiza en dos horas aproximadamente.

[2] Hasta el surgimiento de *Proflow* no existía un *software* específico para esta industria. Las empresas utilizaban *software* de gestión estándar, tipo Tango, sobrellevando sus limitaciones.

> *proponerle al cliente, decirle 'mira: esto es lo que querías y no lo que me proponías al principio'. Entonces ese proceso de conocimiento se da cara a cara con reuniones*[3].

Una vez listo el núcleo del *software*, el proyecto entró en una segunda fase que consistía en el desarrollo de herramientas de monitorización de la producción, con una duración aproximada de un año (los primeros seis meses dedicados al desarrollo intensivo de las herramientas y los siguientes seis meses a los ajustes).

Las dos primeras fases involucraron una inversión de recursos humanos de entre 4 y 10 personas dependiendo del periodo: el periodo que más recursos demandó fue el segundo semestre de 2012 durante el cual trabajaron 10 personas, y en el periodo en que menos trabajadores se involucraron el equipo estaba integrado por 4 personas (segundo semestre de 2013). En promedio, el tamaño del equipo osciló entre 5-6 y 9-10 personas. A lo largo de la trayectoria de *Proflow*, casi todos los empleados de la empresa integraron el proyecto en algún momento. A la fecha del trabajo de campo (junio de 2014) seis personas lo llevaban adelante. Se trata del proyecto de mayor envergadura que la empresa ha realizado desde su nacimiento.

Al iniciar la relación con el cliente se planificó una tercera fase que hasta el momento de elaboración de este artículo no se había llevado a cabo. Dicha fase implicaría incorporar *Machine Learning*, Inteligencia Artificial, Redes neuronales y todos los campos núcleo de conocimiento de la empresa al desarrollo de una herramienta que permitiera medir la "capacidad de carga" del taller.

> *Hoy en día no hay una medida en la industria que a ellos les permita saber cuál es su cuello de botella, dónde están mis mayores problemas [sic], cuántas órdenes debería ser capaz de procesar un operador por día. Esas respuestas, por las características de la industria, hoy en día no están respondidas [sic], no hay conocimiento de eso. Estamos tratando de lograrlo, es complicado, pero estamos tratando. Y esta es la segunda ventaja estratégica o innovadora del sistema, otro de los objetivos que tenía J. cuando quiso empezar este sistema*[4].

Desde la visión de los entrevistados se trataría de la fase más innovadora del proyecto en términos del conocimiento existente en los campos de la computación especialidad de la empresa y su aplicación a la industria de destino. Insumiría alrededor de siete meses y unos cuatro o cinco desarrolladores[5]. La suspensión de esta fase tiene que ver con los acontecimientos que relatamos a continuación.

La empresa cliente fue adquirida por un fondo de inversión, que compró a la vez otras plantas similares en distintas localizaciones de EEUU. De manera que se modificó la organización administrativa y el antiguo dueño (J.) pasó a ser accionista y jefe de producción en la planta original. Los nuevos dueños de la firma realizaron una auditoría del *software* con el objeto de comparar sus prestaciones respecto al *software* genérico utilizado por la competencia y, a partir de este estudio, definieron que *Proflow* era el más adecuado y le dieron continuidad. Por lo tanto, el sistema originalmente diseñado para una sola planta, al momento de realización de las entrevistas debía ser adaptado al uso simultáneo en varias plantas, lo que implica migrar el sistema a otras arquitecturas de *software* y permitirle centralizar toda la información a partir de una tecnología de sistemas distribuidos. De allí que

[3] Entrevista a Patricio D.B., *Project Manager* de *Proflow*, 22/05/2014.
[4] Ibídem.
[5] Su desarrollo efectivo estaba planteado, al momento de realización del relevamiento, para el segundo semestre de 2014.

la tercera fase originalmente ideada quedara postergada por esta última tercera fase real, dado el cambio en las necesidades de negocio del cliente.

- *Organización del proceso de trabajo*

Para el desarrollo y mantenimiento del sistema se lleva a cabo un proceso de ingeniería de *software* iterativo e incremental. El *software* se implementa por partes o períodos de 15 días denominados iteraciones o *sprints*, que el cliente prueba y evalúa. Una vez completada cada iteración, se pasa a la siguiente. No se realiza de una sola vez todo lo que el cliente necesita. Las iteraciones planificadas originalmente pueden variar en tiempo y forma de acuerdo a la marcha del proceso, hay flexibilidad. Las etapas básicas del proceso son: análisis de requerimientos (se establece con precisión y en términos técnicos qué necesita el cliente del *software*), desarrollo, prueba e implementación. Esas actividades se iteran continuamente. Ante el inminente inicio de cada iteración se realiza una estimación colectiva del tiempo que ésta insumirá, a través de una técnica *ad hoc* implementada por la empresa: el *poker planning*[6]. Una vez finalizada cada iteración se realiza una "reunión de prospectiva" en la que se analiza colectivamente qué se hizo, resultados y acciones de mejora constante. En suma, el proceso es incremental porque se va implementando de a poco, y de a poco va creciendo la solución. Para llegar a las soluciones utilizan la dinámica de la tormenta de ideas (*brainstorming)* entre todos los integrantes del equipo.

Para organizar los procesos de trabajo la empresa emplea Metodologías Ágiles tipo *Scrum*[7]. Estas Metodologías involucran el tipo de proceso descripto arriba (iterativo e incremental) y prevén diversos roles y funciones al interior del equipo de trabajo, que en *Proflow* se plasmaron según ilustra el cuadro 1.

A su vez, la empresa ha realizado adaptaciones de la metodología *Scrum* en función de ciertos rasgos de la cultura organizacional de Machinalis que *"chocaban con ese framework"*[8]. Dichas adaptaciones se encuentran principalmente entre los roles y funciones clásicos de los integrantes del equipo de trabajo. En el cuadro 2 pueden apreciarse las modif Los entrevistados aclararon que *Scrum* siempre requiere adaptación, que toda empresa adapta esas buenas prácticas a su realidad.

La gran mayoría de los integrantes del equipo de *Proflow* han tenido/tienen título universitario o estudios superiores avanzados[9]. En las tareas de desarrollo predominan los profesionales de Ingeniería o Licenciatura en Computación, y en las tareas más ligadas al *management* los Ingenieros en Sistemas. Los no graduados tienen el mismo tipo de participación en el proyecto; en este sentido es política de la empresa que la titulación no resulte determinante en la asignación de tareas ni responsabilidades, en general ello tiene más que ver con la edad y la experiencia que con el grado de calificación formal[10].

- *Aprendizajes, desarrollo de capacidades y vinculaciones*

Los entrevistados destacaron que el efectivo desarrollo de *Proflow*, dada su envergadura y la inexperiencia previa de la empresa en la industria de destino del sistema, fue posible, por

[6] Mediante naipes especialmente numerados, cada integrante del equipo de desarrollo propone una estimación. Cuando las estimaciones tienden a coincidir se establece un promedio colectivo, y si existen diferencias significativas se discute y a partir de las visiones planteadas se realiza nuevamente el juego.
[7] Ver Takeuchi y Nonaka (1986) y Sutherland y Schwaber (2012).
[8] Entrevista a Patricio D.B., *Project Manager* de *Proflow*, 24/06/2014.
[9] Ingenieros en Computación o en Sistemas, Licenciados en Computación, estudiantes avanzados de esas carreras.
[10] Ninguno de los dos entrevistados del proyecto, por ejemplo, estaba graduado al momento de asumir sus puestos actuales en el proyecto (*Project Manager* y *Technical Leader*).

un lado, gracias a un cúmulo de capacidades preexistentes, relacionadas principalmente con las competencias de los recursos humanos en las áreas de desempeño de la empresa.

> *Somos una empresa que se especializa en Python y Django, entonces acá hay gente reconocida mundialmente por su trabajo en Python y por su contribución a la comunidad de Python*[11].
>
> *Para nosotros tener científicos dando vuelta por acá es habitual. Si no tengo el científico acá no puedo seguir*[12].

CUADRO 1: IMPLEMENTACIÓN DE METODOLOGÍAS ÁGILES (*SCRUM*) EN PROYECTO *PROFLOW*

Rol	Requerimientos de rol	Implementación en *Proflow*
Dueño del producto (*Product Owner*)	Define especificaciones de producto y prioridades.	Cliente
Project Manager	- Gestiona el proyecto. - Se comunica con el *Product Owner*. - Estructura los requerimientos del *software* y los comunica al equipo de desarrollo. - Toma decisiones de organización del equipo. - Coordina reuniones. - Registra las tareas y estimaciones que surgen en el sistema de gestión de proyectos de la empresa (herramienta de gestión de proyectos denominada JIRA[13]). - Oficia de nexo entre el cliente y el equipo de desarrollo y entre éste y las áreas de la empresa que lo requieran. - Cumple una función de "protección" del equipo de las presiones externas para que los desarrolladores puedan trabajar con tranquilidad en sus tareas específicas[14]. - No requiere necesariamente el mismo nivel de competencias técnicas que el resto del equipo, sino más bien de competencias de gestión que aporten dinamismo al proceso de desarrollo (como capacidades de comunicación, manejo de grupos, administración de tiempos de trabajo).	Definido *ad hoc* por la empresa
Technical Leader	- Detecta riesgos tecnológicos. - Eventualmente diseña la arquitectura del *software*. - Toma decisiones relevantes a nivel tecnológico pero a partir de soluciones propuestas por el equipo de desarrollo, que en general se consensuan.	Definido *ad hoc* por el equipo de desarrollo
Equipo de desarrollo	- Asigna las tareas internamente. - Desarrolla el sistema y codifica las pruebas automáticas. - Diversos roles de desarrolladores: *Front-End*, *Back-End*, Administrador de Base de Datos, *Sysadmin*, *Tester*.	Definido *ad hoc* por la empresa según especificidades de los proyectos

Fuente: Elaboración propia

[11] Entrevista a Patricio D.B., *Project Manager* de *Proflow*, 22/05/2014.
[12] Ibídem.
[13] Es una herramienta de *management* de proyectos. Más información en https://www.atlassian.com/es/*software*/jira
[14] Más adelante volveremos sobre este punto.

CUADRO 2: ADAPTACIONES DE LA METODOLOGÍA *SCRUM* A LA CULTURA ORGANIZACIONAL DE MACHINALIS SEGÚN ESPECIFICIDAD DEL PROYECTO *PROFLOW*

Rol	Requerimientos de rol 'canónicos' en *Scrum*	Adaptaciones	Fundamentos
Dueño del producto (*Product Owner*)	Tiene contacto directo con el equipo de desarrollo.	- Sólo el *Project Manager* habla con el equipo, dado que el *Product Owner* es el cliente. - Se utiliza el criterio canónico cuando el *Product Owner* tiene un escaso conocimiento sobre *software*[15] y es necesario un nexo que garantice la idoneidad de las decisiones técnicas.	*"El Project Manager trata de proteger un poco al equipo para que el equipo esté tranquilo (...). Regula el flujo de información y conecta con el equipo y controla (...). Por ahí porque la cultura de Machinalis es una cultura mucho más técnica, que no tienen por ahí las habilidades blandas muy desarrolladas, por decirlo así, entonces por ahí les cuesta hablar con el cliente* [sic]"[16].
Project Manager	No es el responsable máximo del proyecto: el equipo se auto-organiza y existe una figura de facilitador (*Scrum Master*[17]) que ayuda a la eficiencia pero no toma decisiones respecto a los demás integrantes.	Es el responsable máximo del proyecto: toma decisiones en el marco de una organización del equipo, donde predomina el consenso.	*"El Project Manager es a la vez analista funcional, es como el primer analista funcional, después una vez que ése hizo el primer análisis de requerimientos por ahí lo suelta al equipo y dice 'bueno, tenemos que hacer esto y éstos son más o menos los requerimientos', y por ahí al equipo se le ocurre una mejor forma de hacerlo, pero sí, el Project Manager es quien dentro de su rol es analista funcional, que es tratar de dilucidar qué quiere el cliente y acomodar las ideas para presentárselo al equipo"*[18].
Technical Leader	Es quien más conocimiento técnico posee.	Es quien más experiencia tiene sobre el proyecto.	"[En las metodologías ágiles se pensaba al *Project Manager* y al *Technical Leader* como] *'Gurúes que saben mucho'. Pero en esta empresa hay mucha gente que sabe mucho. Después hay proyectos donde todos saben más o menos lo mismo, entonces hay proyectos como el nuestro donde el rol del Technical Leader se decide por cuánto* ***más conoces del proyecto*** *"*[19].
Equipo de desarrollo	- Tareas rígidas por integrante. - Equipos estables que desarrollan todos los proyectos	- Premisa de *"equipo multifuncional donde todos hacen todo"*[20], sin diferenciaciones rígidas con el objeto de evitar la interrupción de tareas. - Equipos variables conformados *ad hoc* según las especificidades de cada proyecto.	*"Perdemos un poco de especialización, no siempre la interfaz gráfica es bonita y muchas veces el código back-end no es prolijo, pero la gestión y el desarrollo es mucho más fluido. Eliminamos interdependencias"*[21].

Fuente: Elaboración propia

[15] Las Metodologías Ágiles *Scrum* presuponen lo contrario.
[16] Ibídem.
[17] Sutherland y Schwaber (2012).
[18] Entrevista a Patricio D.B., *Project Manager* de *Proflow*, 24/06/2014.
[19] Entrevista a Patricio D.B., *Project Manager* de *Proflow*, 24/06/2014.
[20] Ibídem.
[21] Entrevista a Patricio D.B., *Project Manager* de *Proflow*, 24/06/2014.

El *Project Manager* señala estos elementos como diferenciales de Machinalis respecto a otras empresas de *software* de Córdoba.

En visión retrospectiva los entrevistados señalan una variedad de aprendizajes derivados del proyecto, asociados principalmente con aspectos organizacionales y en menor medida comerciales.

En relación con el primer aspecto se destacan, por un lado, las adaptaciones de las Metodologías Ágiles realizadas durante el desarrollo del proyecto que ya mencionamos en el apartado anterior, fruto de las dificultades de organización con las que el equipo se fue encontrando.

> *Cuando se comenzó con este sistema, como que no estaba muy claro estas metodologías ágiles [sic], entonces había una persona que era el líder técnico, y además era Project Manager, además hablaba con el cliente, además era desarrollador, hacía todo, y duró cuatro meses. Entonces creo que un aprendizaje mayor fue cuando se ingresó un Project Manager al proyecto. (…) Ése fue el principal punto de inflexión, cuando se dieron cuenta que había un rol que no era técnico que servía. (…) Incluso después de ver en Proflow que el rol de Project Manager funcionaba y era crítico se creó una política organizacional entre comillas de decir 'todo proyecto tiene que tener un Project Manager, nunca más ponemos a un desarrollador al frente del cliente'*[22].
>
> *Ahora el otro aprendizaje grande que está teniendo Proflow es que el rol del Project Manager está muy saturado. Hoy en día tiene muchísimas tareas para hacer, entonces están empezando a ver dentro del rol de Project Manager cuáles son los roles [sic] y empezar a dividirlo en distintas personas*[23].

Además, los entrevistados destacan que a partir del desarrollo de *Proflow*, por su envergadura, la empresa aprendió a reconocer las potencialidades y límites del tamaño del equipo de desarrollo.

> *Creo que de eso está bueno aprender, hasta dónde puede llegar el equipo, qué tan grande puede ser el equipo, porque tampoco puedes tener tantos desarrolladores, si tienes muchos desarrolladores empiezas a tener mucha incertidumbre, y la iteración va a abarcar más tareas, y muchas tareas son interdependientes (…), qué tan larga puede ser una iteración, qué tan malo puede ser que se extienda. (…) Por ahí el cliente te pide capacidad de desarrollo y vos decís 'uno más, uno más', y llega un momento en que con ese uno más termina todo así [dado vuelta]*[24].
>
> *Cuando llega ese límite, otro equipo. Hay que hacer otro equipo con las mismas características. Los dos equipos trabajan sobre la misma pila de tareas pero se organizan independientemente. (…) A las dos semanas se vuelven a juntar, y se separan, y así*[25].

Otro de los aprendizajes organizacionales relevantes derivó del hecho de que el equipo determine la duración de las iteraciones según sus capacidades y ya no el cliente, a partir de haber aceptado una imposición del cliente al respecto que resultó en un fracaso para el avance del proyecto[26]. Esto se estableció como una regla en la empresa a partir de allí. Al mismo

[22] Entrevista a Patricio D.B., *Project Manager* de *Proflow*, 24/06/2014.
[23] Ibídem.
[24] Entrevista a Francisco, *Technical Leader* de *Proflow*, 24/06/2014.
[25] Entrevista a Patricio D.B., *Project Manager* de *Proflow*, 24/06/2014.
[26] Relatan que el cliente pidió *sprints* de 45 días, lo que generó que las desviaciones en las tareas respecto a las estimaciones fueran altísimas. *Sprints* de 15 días permiten a la empresa tener una mejor revisión de las estimaciones.

tiempo, este aprendizaje de tipo organizacional condujo a establecer modificaciones en la gestión de los proyectos desde el área comercial: la propuesta circuló desde el equipo de *Proflow* al área comercial de la empresa, y a partir de ese momento, toda vez que esta última área negocia con el cliente acuerda específicamente que este último no va a tomar decisiones sobre el esquema de iteraciones.

Los entrevistados destacan que el proceso permanente de ensayo y error es una de las fuentes principales del tipo de aprendizajes mencionados, y que ello es propio, a su vez, de las Metodologías Ágiles que implementan para organizar el trabajo al interior de los proyectos.

> *Básicamente somos una empresa que es nueva, entonces estamos todo el tiempo chocándonos contra la pared, siempre cometemos errores y tratamos todo el tiempo de tratar de ver cuáles son los errores que cometimos y tratar de ver de que la empresa en futuros proyectos no vuelva a cometer los mismos errores*[27].

En relación con esto último, los entrevistados enfatizan que este tipo de metodologías de desarrollo de *software* también permiten aprender al nivel de los procesos empresariales en general. En este sentido relataban, al momento de realización del trabajo de campo, que se encontraban desarrollando un proyecto interno que reunía a todas las áreas de la empresa, destinado a reducir el tiempo global de desarrollo en base a los aprendizajes surgidos de todos los proyectos realizados.

Una fuente muy clara de aprendizajes y desarrollo de nuevas capacidades en la empresa a partir de *Proflow* es la vinculación con el cliente. Éste parece haber tenido en este caso un papel clave en las condiciones para innovar en el desarrollo del *software*, principalmente por tratarse de una empresa también innovadora al interior de su rama industrial.

> *Entonces no había en el mercado un software ajustado a esta industria. La competencia utiliza este otro sistema, que no tiene las prestaciones de Proflow, pero como el estándar sabe que medianamente funciona, lo usa. (…) Pero J., que es un poco más visionario, quería algo que le dé una ventaja estratégica un poco más importante, entonces quería desarrollar uno que le quedara como guante a la industria*[28].

También es importante señalar que, desde la visión de los entrevistados, esta vinculación fue clave por haberse producido una dinámica de aprendizajes e innovaciones en la que ambas empresas adquirían mayor protagonismo en diversas etapas del desarrollo del proyecto según las necesidades y conocimientos que cada una manifestaba y podía aportar. Al mismo tiempo, destacan que esta dinámica permite al cliente innovar en sus propios procesos y correr el límite de sus propios conocimientos.

> *Básicamente, todas estas nuevas herramientas cambian completamente la forma de trabajo de ellos. Entonces es un trabajo que tienes que hacer en conjunto con el cliente porque vos puedes innovar para la parte de software, pero vos necesitas sí o sí alguien del lado del otro que identifique como estás innovando con el software y te diga 'entonces yo puedo innovar de esta forma en mis procesos'. Así que sí, es una innovación por ambos lados*[29].

A su vez, el tipo de relación establecida con el cliente y el devenir del proyecto[30] implicaron la necesidad de establecer vinculaciones con otros agentes que constituyeron fuentes de conocimiento externas de relevancia. Respecto al desarrollo de herramientas para

[27] Entrevista a Patricio D.B., *Project Manager* de *Proflow*, 24/06/2014.
[28] Entrevista a Patricio D.B., *Project Manager* de *Proflow*, 22/05/2014.
[29] Ibídem.
[30] En concreto, la suspensión de la tercera fase originalmente planteada y su reemplazo por otra con características diferentes.

medir la capacidad de carga del taller, si bien fue suspendido por el cliente, al momento de realizar el trabajo de campo Machinalis continuaba trabajando en esta dirección independientemente del cliente aunque bajo su conocimiento, y para ello se encontraban en vinculación con un espacio académico especializado del extranjero.

> *Machinalis se puso en contacto con el centro de modelado matemático del laboratorio de análisis combinatorio de la Universidad de Chile, porque ellos son personas que se especializan en modelar procesos productivos. Y ver si entre ellos [el cliente], nosotros y el laboratorio de Chile somos capaces de armar algún modelo matemático que nos permita optimizar eso. (…) Y ese fue el momento más complicado, desde que yo estuve, porque ahí es donde más tuvimos que innovar, ahí es donde más tuvimos que salir a decir 'tenemos estos problemas, tenemos esta gente, tenemos esta herramienta, esto tiene que seguir andando, muchachos ¿cómo hacemos para seguir andando? Bueno… llama a la universidad de Chile'*[31].

El CEO de Machinalis decidió iniciar la relación con el laboratorio chileno sin tener muchas certezas sobre los resultados: *"Al menos servirá para probar y decir 'por ahí no es'"*[32].

Además del laboratorio de Chile, Machinalis se encontraba en conversaciones con una empresa de Dallas-EEUU especializada en sistemas distribuidos, buscando que ésta les proveyera el soporte para poder diseñar la arquitectura del nuevo *software* multi-plantas[33].

Son muy frecuentes e intensas también las vinculaciones de Machinalis con las comunidades tecnológicas de referencia del *framework* de desarrollo (Django[34]) y el lenguaje de programación en que se especializan (Python[35]), lo que se manifestó con claridad durante el desarrollo de *Proflow* aún dado el perfil comercial de proyecto.

Una última fuente de aprendizajes que destacan los entrevistados es la aplicación de normas de calidad ISO 9001 en los procesos de gestión.

> *Esta certificación se realizó cuando el proyecto llevaba poco más de un año. Lo valioso de la certificación es que nos permite detectar oportunidades de mejoras y mejora continua a los procesos de gestión y procesos de desarrollo del proyecto. (…) Hoy en día el proceso de gestión de proyectos de la empresa está muy embebido de experiencias y prácticas realizadas en el proyecto [Proflow]*[36].

- *Resultados de innovación y medidas de performance.*

La medición de la *performance* del proyecto es interna al equipo de desarrollo, los integrantes desconocen cómo impacta el proyecto en la productividad y el desempeño económico de la empresa. Cada vez que comienza una iteración se elabora una lista de tareas

[31] Entrevista a Patricio D.B., *Project Manager* de *Proflow*, 22/05/2014.
[32] Ibídem.
[33] Al momento de elaboración de este artículo dicha vinculación estaba en sus comienzos por lo que no contamos con mayor información ni evaluaciones al respecto.
[34] Un *framework* es una estructura conceptual y tecnológica de soporte para la organización y desarrollo de *software*. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado, entre otras herramientas. Django es un *framework* de desarrollo web de código abierto, escrito en el lenguaje de programación Python, que respeta el paradigma conocido como *Model Template View*. La meta fundamental de Django es facilitar la creación de sitios web complejos, poniendo énfasis en el re-uso, la conectividad y extensibilidad de componentes, el desarrollo rápido y el principio "no te repitas" (DRY, del inglés Don't Repeat Yourself). Más información en https://www.djangoproject.com/.
[35] Python es un lenguaje de programación con licencia de código abierto, compatible con la licencia pública general de la GNU. Más información en http://www.python.org/.
[36] Entrevista a Patricio D.B., *Project Manager* de *Proflow*, 24/06/2014.

a realizar y cada tarea conlleva una medida de esfuerzo, que en el caso de *Proflow* se traduce en tiempo (horas y días). Para monitorizar su cumplimiento post-iteración utilizan un gráfico llamado *burndown*, propio de las Metodologías Ágiles, que permite medir la productividad dentro de la iteración: en concreto, la distancia existente entre lo efectivamente realizado y la velocidad ideal de trabajo del equipo. A nivel del proyecto en general aplican la misma medida pero en términos "macro": trabajan por objetivos antes que por tiempos y realizan un seguimiento continuo muy similar de su cumplimiento.

La cuestión de la medición de la *performance* es relevante para la empresa, en relación, a su vez, con el tipo de organización del trabajo que practican:

> *El gran problema que tenemos en el software es la estimación. El cliente viene y te dice 'cuánto me lleva hacer esto', y no hay forma, en software es casi imposible de determinar. Entonces vos nunca te comprometes a más de dos semanas*[37].

> *Hay como una sobrecarga en la estimación. Te lleva tiempo. Por ahí de diez días, capaz que medio día nos toma estimar las tareas*[38].

Los integrantes del proyecto sí acceden a información de *performance* del cliente, que les provee el propio *software*, y de este modo la empresa puede seguir y evaluar progresivamente qué cambios ha producido el sistema en relación con sus funcionalidades. Según los entrevistados, dichos cambios se producen a nivel de la productividad y también en términos de la cultura organizacional de la empresa cliente.

Los resultados de innovación efectivamente obtenidos durante el desarrollo de *Proflow* reúnen principalmente: a) las innovaciones organizacionales ya referidas, b) el sistema de gestión en sí, y c) innovaciones técnicas que forman parte del producto pero no son necesariamente visibles aunque sí se socializan en las comunidades tecnológicas de pertenencia.

> *Creo que somos de alguna forma pioneros en atacar el problema de este tipo de industria específica, que es orientada al shop, una línea de producción de cosas que entran por allá y salen por acá*[39].

En particular, señalan que lograron realizar las dos primeras fases del proyecto gracias al desarrollo de un algoritmo de árboles de decisión. Según explican, ello constituye una innovación técnica:

> *Es innovarlo… estás innovando porque lo estás utilizando para esta industria, en cierto punto. O sea, los algoritmos de árboles de decisión existen, pero haberse dado cuenta de que un algoritmo de árbol de decisión le iba a permitir a este Planificador, en base a todas las características de esta industria, poder hacer una decisión rápida de qué proceso hacer con la pieza, eso no, no existía antes, entonces Machinalis se dio cuenta de esos nexos, por decirlo así*[40].

En segundo lugar comentan que un proceso similar ocurrió en torno a las tecnologías que utilizan para desarrollar el *software*: por un lado el *framework* web Django que combinan con otras tecnologías que suplen las herramientas que van faltando.

> *Nosotros expandimos, llegamos a los límites de esta herramienta. (…) Tuvimos que mover la línea de esa herramienta. (…) Entonces hoy en día no hay un producto hecho así con Django. (…) Y hoy en día estamos utilizando, no sé, ocho tecnologías distintas*

[37] Ibídem.
[38] Entrevista a Francisco, *Technical Leader* de *Proflow*, 24/06/2014.
[39] Ibídem.
[40] Entrevista a Patricio D.B., *Project Manager* de *Proflow*, 22/05/2014.

para hacer esto. (...) Y todas estas, decimos, de alguna forma meterlas adentro de Django, y eso es bastante innovador en cuanto a Django[41].

Por otra parte introdujeron, a partir de la combinación de distintas tecnologías, una innovación útil para lograr extraer un gran volumen de información del sistema por parte de muchos usuarios simultáneamente, problema que no estaba resuelto previamente y resultaba imprescindible para determinadas operaciones en la planta.

Cosas que antes hubieran sido imposibles, al menos según lo que estaba en la red y lo que fuimos encontrando, le dimos esa vueltita de rosca. (...) Juntamos cosas que no estaban juntas antes (...). Eso nos habrá llevado más o menos un mes, pero la verdad que tuvimos resultados muy muy buenos [sic]. (...) Nosotros trabajamos con modelos. Los modelos son todos conocidos... quizá en los que fuimos originales nosotros es en usarlos para esta industria, para este problema en particular[42].

El *Project Manager* explicó que a ese problema la comunidad mundial de Django no lo había solucionado. Por último, destacan que introdujeron también innovaciones técnicas en el sistema operativo Linux para resolver un problema operativo del taller (necesidad de uso de "terminales bobas"[43]).

Dentro del universo de posibilidades que había acá, seleccionamos un par que nos servían, las mezclamos, las combinamos, y las añadimos a Django. De hecho Django, que era un framework de desarrollo web que no estaba preparado para recibir terminales bobas, de repente nosotros juntamos unas cuantas herramientas que daban vueltas por la red y toqueteamos el sistema operativo, logramos que el sistema operativo dijera 'dame lo que quiero'[44].

El *Project Manager* explicó que esa innovación puede ser aplicable a cualquier sistema (por ejemplo, el sistema de gestión de un hospital).

Por último, respecto a las posibilidades de liberar los resultados de innovación obtenidos durante el desarrollo de *Proflow*, hay evidencia de cierta tensión entre la propiedad privada del *software* cuando el proyecto es para un cliente, y el impulso a la innovación que motiva a la empresa. Cuando se realiza un desarrollo a medida para un cliente, la propiedad intelectual, por contrato, pertenece a este último. De todas maneras, cuando existen innovaciones relevantes que no involucran la esencia del *software*, Machinalis tiende a liberarlas entendiendo que de ese modo se retribuye a su vez a la comunidad tecnológica por herramientas libres que también utilizan a beneficio del cliente sin que éste las pague o siquiera las conozca.

3.2. Un Proyecto Interno: Quepy

Quepy es un *framework* de *Python* para transformar preguntas realizadas en lenguaje natural a preguntas en lenguaje de consulta de una determinada base de datos, es decir, un *software* conversor de preguntas en lenguaje natural a consultas de bases de datos. El objetivo del programa es que con poca codificación el usuario pueda construir su propio sistema para acceder a partir del idioma natural a su base de datos. Si bien existen en el mercado productos similares que tienen una antigüedad de alrededor de cinco años (como Siri y Google Now)

[41] Ibídem.
[42] Entrevista a Francisco, *Technical Leader* de *Proflow*, 24/06/2014.
[43] Se trata de computadoras que permiten mostrar información y operar como usuario de un sistema determinado sin acceder al servidor ni al sistema operativo ni a la posibilidad de utilizar otros programas en la misma terminal (por ejemplo, las ubicadas en la recepción de un hospital o un punto de venta de un comercio). Es decir, sólo devuelven información ante instrucciones delimitadas.
[44] Entrevista a Patricio D.B., *Project Manager* de *Proflow*, 24/06/2014.

estos productos son de código cerrado. Quepy persiguió ubicarse en este nicho de productos, pero constituyendo la única pieza aplicada de tipo *open source*.

- *Origen y fases de desarrollo del proyecto*

En el origen del proyecto confluyeron tanto una necesidad estratégica definida por la empresa, como el devenir de un proyecto anterior denominado SOSE (*Semantic Ontology Search Engine*).

En 2012 la empresa definió una estrategia de negocios consistente en posicionarse en el mercado a través del desarrollo y posterior liberación de un producto *open source* puramente innovador. Así, surgió Quepy:

> *Por propósitos estratégicos, y la iniciativa de Marcos en su rol de CEO, para posicionar a la empresa, se decide hacer un 'proyecto innovador', para que los otros sepan que nosotros sí lo podemos hacer y nos quieran contratar; una estrategia publicitaria, digamos, con un plus adicional que viene por el lado de que a la gente de esta empresa le gusta hacer proyectos que entran en la categoría de open source. Entonces, a los empleados les gusta, da visibilidad fácil (los proyectos open source se difunden fácil en internet si son buenos). (…) Vamos a hacer algo que sea innovador, y que nos de visibilidad, y que mantenga a la gente contenta*[45].

La apuesta por un producto *open source* se justificaba, además, en que éste ofrecía la posibilidad de exponer públicamente toda la mecánica interna desarrollada, de mostrar un grado elevado de calidad interna de los productos de la empresa, siendo además de rápida y fácil visibilidad. Se buscaba un producto que oficiara como "carta de presentación" de las capacidades de desarrollo de la empresa:

> *El software open source es algo que puede verlo todo el mundo. Y no me refiero a la calidad del "cascarón" del producto, sino lo que está adentro. (...) Si fuera un reloj, y lo abrís, los engranajes tienen que ser dorados, tiene que estar precioso por dentro, porque esa es tu carta de presentación para la empresa que te quiere contratar*[46].

Con la misma motivación surgió el proyecto anterior, SOSE. Dicho proyecto, técnicamente, fracasó. Se trataba de un proyecto más ambicioso que Quepy, con la finalidad de desarrollar un *software* que permitiera la carga de bases de datos desde lenguaje natural y que a la vez permitiera, a través de procesamiento de lenguaje natural, extraer y recomendar información desde esas bases de datos. Estos eran los objetivos "duros" del proyecto, a los cuales se sumaba un "objetivo suave" de posicionarse en la comunidad Python. Los objetivos duros no se lograron nunca con un nivel de calidad interna aceptable para la comunidad. La comunidad Python funcionó como testeador y validador de los avances del proyecto. Durante el proceso de desarrollo de SOSE, miembros de la comunidad Python señalaron que dada la complejidad con la que se estaba manejando Machinalis, uno de los objetivos intermedios del proyecto estaba obstaculizando el desarrollo. Quepy surgió de ese punto del proceso, de la idea de extirpar ese objetivo intermedio y hacer una pieza más simple y pequeña. De este modo nació Quepy como un proyecto *"más minimalista, menos ambicioso, pero mucho más concreto"*[47], que es una derivación de SOSE y del conocimiento preciso de los aspectos que no funcionaron en aquel proyecto originario.

El equipo de Quepy se conformó con integrantes del proyecto SOSE. Un *Project Manager*, dos desarrolladores y una consultora desarrolladora. El *Project Manager* (Elías A.),

[45] Entrevista a Rafael C., desarrollador de Quepy e impulsor de la idea del proyecto, 28/05/2014.
[46] Ibídem.
[47] Ibídem.

quien gestionó el proyecto, fue analista funcional *part time* en SOSE y fue estudiante de Ingeniería en Computación (UNC). Entre los desarrolladores "puros" estuvieron Rafael C. (Licenciado en Ciencias de la Computación, FAMAF, UNC), docente especializado en *Natural Language Processing* con experiencia en programación de experimentos en este campo, y "John" (programador sin experiencia académica, pero con una alta capacidad para comprender las ideas intuitivas que se le trasmiten y llevarlas a la programación, según los entrevistados). El equipo de desarrollo se completaba con Laura A. (Dra. en Lingüística Computacional, Universitat de Barcelona), que reviste la figura de consultora-desarrolladora en el área de *Natural Language*, pero que no participó específicamente programando. Por último, existe un reconocimiento especial y simbólico dentro de la empresa a un integrante (Horacio D.) que participó de la idea en SOSE que dio origen al proyecto, pero que no programó específicamente en Quepy.

El desarrollo del proyecto puede dividirse en tres etapas: la primera tuvo lugar entre junio y octubre de 2012, la segunda entre diciembre de 2012 y marzo de 2013, y la última etapa entre julio y septiembre de 2013.

La primera etapa consistió en el desarrollo de la primera versión de Quepy para generar consultas de base de datos en lenguaje SPARQL. La definición de utilizar este lenguaje por parte del equipo constituyó un hito en la primera semana del proyecto, que facilitó el resto del desarrollo e implicó capacitación y formación del equipo en el uso del mismo. Se buscó que la primera versión permitiera consultas en la base de datos de DBpedia (base formada por contenidos de Wikipedia).

Durante la segunda etapa se desarrolló un sitio web demo de Quepy y se trabajó en la generación de documentación y en el agregado de la cantidad de preguntas reconocidas. Al interior de esta etapa los entrevistados identificaron dos hitos del proceso de desarrollo del proyecto. Por un lado, a partir de la producción de la demo de Quepy se extendió esta práctica al resto de los proyectos y la empresa definió adoptarla como política de difusión y presentación. El otro hito fue la liberación del código de Quepy por versiones.

> *Llegamos hasta un hito y a partir de ese hito liberamos la versión 01 de Quepy (…), establecemos una línea base (…), lo primero que vamos a mostrar de Quepy llega hasta acá, y ahí mostramos. Luego hacemos más mejoras, cuando tenemos tiempo armamos un sprint y podemos sacar una versión 02. Y así vamos liberando el código de Quepy*[48].

La última etapa del proyecto consistió en ampliar las consultas de Quepy al lenguaje MQL, para poder generar consultas en una de las más grandes bases de datos que existen en la actualidad, Freebase[49]. Ello implicó capacitación y formación del equipo en el lenguaje MQL y en la estructura y manejo de la base de datos Freebase.

- *Organización del proceso de trabajo*

Uno de los aspectos en que este proyecto resultó particularmente dinámico fue en las innovaciones en la organización del proceso de trabajo. Para el desarrollo de Quepy también se implementaron Metodologías Ágiles *Scrum* realizando adaptaciones acordes a diversas particularidades del proyecto.

[48] Entrevista a Elías A., *Project Manager* de Quepy e impulsor de la idea del proyecto, 18/06/2014.

[49] Freebase es una base de datos colaborativa que compuesta por metadatos en línea obtenidos a partir de múltiples fuentes, incluyendo contribuciones "wiki" individuales, creada por Metaweb, que fue adquirida por Google en 2010. Los datos de Freebase están disponibles para uso free/libre, comercial y no comercial bajo una licencia *Creative Commons*.

Quepy funcionó a través de *sprints* de alrededor de 15 días. Al inicio y al final de cada *sprint* se realizaban reuniones de revisión y planificación donde se definía el alcance (*scope)* del proyecto y del *sprint*, las metas, las prioridades, la planificación de las tareas necesarias para alcanzar cada meta, se estimaba y asignaba un tiempo de trabajo por tarea y se evaluaba la consecución de las distintas tareas. Así, el proceso productivo se organizó a través de una sucesión de *sprints*, donde quedaban "congeladas" las tareas correspondientes al período. En el marco de cada *sprint* se mantenían breves reuniones diarias de sincronización (de un máximo de 10 minutos) llamadas *standups*, donde se dialogaba acerca de qué se había hecho, qué se estaba haciendo y qué impedimentos habían aparecido. De haberlos, el *Project Manager* podía hacer algo al respecto y al mismo tiempo el resto del equipo quedaba notificado a partir de la reunión.

El *Project Manager* de Quepy explicó en forma estilizada la dinámica de los tres tipos de reuniones que se realizaban para cada *sprint*: una reunión inicial (de *brainstorming)*, una reunión de planificación (*planning)* y luego reuniones regulares y sucesivas de revisión. En la inicial se define el alcance del proyecto y el problema a abordar, y se articula la primera idea genérica del proyecto. Allí tienen gran participación los desarrolladores y, de haberlos, consultores especialistas. Aparecen lo que denominan *spikes:* "cosas para investigar" cuando no se pueden estimar los tiempos de las tareas o las tareas mismas[50]. Ante estas situaciones se plantea: *"Esto no lo puedo estimar, no sé si lo podríamos hacer así. Hay gran incertidumbre, entonces se dedica un tiempo para investigar esa línea"*[51]. Luego el equipo vuelve con los resultados de sus *spikes* a una reunión de planificación. Allí se busca concretar la identificación de tareas y una estimación de resultado e insumo-tiempo asociados, lo que permite planear las iteraciones para alcanzar metas y "entregables". Se especifican también las tareas que no se pueden estimar y esto se tiene en cuenta para definir los plazos de entrega (*deadlines)*. En los proyectos de desarrollo internos no monetizados, se pueden dejar tareas sin estimar y crear tareas a mitad de una iteración e, incluso, redefinir su alcance, lo que se sale de un esquema de *Scrum* canónico. Por último, se realizan reuniones de evaluación de lo realizado en cada iteración y se revisan las tareas para la siguiente. Se fija una fecha para el primer *release:* una versión de lanzamiento del producto. Si se cumplió con todos los entregables pre-establecidos, hay *release;* en caso contrario se evalúan las razones del incumplimiento y se re-planifica. El proceso general continúa hasta alcanzar cierto umbral de calidad/resultados que dé lugar a un producto determinado satisfactorio para la empresa. La organización del trabajo a través de *sprints*, reuniones de planificación y *standups* es propia de las Metodologías Ágiles que se implementan en la empresa.

En relación con los roles y funciones al interior del proyecto, para el desarrollo de Quepy se realizaron también adaptaciones de los criterios clásicos de *Scrum*, como se detalla en el cuadro 3: algunos roles fueron creados para el proyecto (consultora-desarrolladora), otros redefinidos o modificados en sus funciones y otros mantenidos.

Otra de las principales adaptaciones, producto de la naturaleza interna del proyecto, es la consideración de tareas sin estimar en las reuniones de planificación que describimos anteriormente. Esto implica la aceptación del hecho de que en un proyecto puramente innovador los resultados son altamente inciertos.

Por último, se realizaron adaptaciones operativas en relación con el grado de periodicidad de las reuniones de trabajo vinculadas a la disponibilidad horaria de los miembros del equipo.

[50] Entrevista a Elías A., Project Manager de Quepy e impulsor de la idea del proyecto, 18/06/2014.
[51] Entrevista a Elías A., *Project Manager* de Quepy e impulsor de la idea del proyecto, 18/06/2014.

CUADRO 3: ADAPTACIONES DE LA METODOLOGÍA *SCRUM* A LA CULTURA ORGANIZACIONAL DE MACHINALIS SEGÚN ESPECIFICIDAD DEL PROYECTO QUEPY

Rol	Requerimientos de rol 'canónicos' en *Scrum*	Adaptaciones
Consultora – desarrolladora	No existe	Creación del rol para incorporar a experta lingüística computacional. Su participación era activa en reuniones iniciales, de planificación y reuniones de replanteo del proyecto o re-diseño del producto.
Product Owner	Define especificaciones de producto y prioridades. Habitualmente es el cliente.	- El proyecto interno no monetizado implica inexistencia de clientes. - Agentes que definieron características de la herramienta: consultora, CEO y desarrolladores que colaboraron con la idea (tanto los que formaron parte del equipo de Quepy como del proyecto anterior).
Project Manager	*Scrum Master*	No se generaron.
Technical Leader	Quien más conocimiento técnico posee.	No existió: equipo pequeño donde las decisiones técnicas se tomaban en conjunto entre los desarrolladores y por consenso.
Equipo de desarrollo	Tareas rígidas por integrante.	Multifuncionalidad.

Fuente: Elaboración propia

- *Aprendizajes, desarrollo de capacidades y vinculaciones*

El proyecto partió de un conjunto de capacidades pre-existentes en el área de *Natural Language Processing* de dos de los integrantes del equipo. Además, contaban con el funcionamiento comprobado de todo el equipo en el proyecto anterior, aunque la mayoría de los integrantes tenía poca práctica en el proceso de desarrollo de *software* de la empresa (gestión de proyectos, idiosincrasia de la organización, etc.). Esto es, partían de un nivel importante de conocimiento de tipo codificado proveniente de su educación formal, pero menos *know how* en la producción de *software*.

En visión retrospectiva, los entrevistados señalaron una variedad de aprendizajes derivados del proyecto, concentrados en gran medida en materia organizacional.

Uno de los primeros aprendizajes del proyecto (surgido en el primer *sprint*) fue el reconocimiento de la necesidad de gestionar los proyectos internos con el mismo nivel de profesionalidad que los proyectos para clientes, con adaptaciones específicas. La organización del proceso de trabajo en SOSE fue más rudimentaria que en Quepy, ya que se trató de uno de los primeros proyectos internos (el proceso de las reuniones era desorganizado y utilizaban otro sistema de gestión de proyectos menos eficiente antes de implementar JIRA). Durante el desarrollo de Quepy comenzaron a aplicar modalidades de gestión de proyectos para clientes adaptadas a las particularidades del proyecto interno y a los cambios generales en la empresa.

Uno de los aprendizajes más importantes que surgió de Quepy fue el reconocimiento de que este tipo de proyectos (internos) necesariamente implica tareas y logros que no se pueden estimar y que su desarrollo puede llevar una cantidad de tiempo incierta, sin converger necesariamente en un resultado útil. Esto es debido,

> *al hecho de que algunas de esas tareas tienen una altísima incertidumbre, es decir, en un proyecto de innovación decimos 'esto puede demorar esto', pero no sabemos*

realmente. Es algo innovador, y entonces no podemos estimar. Esto es una de las lecciones aprendidas que nos han dejado estos proyectos[52].

Este hecho implicó un aprendizaje para todos los niveles de la organización: a los mandos altos y medios les permitió reconocer y aceptar esta particularidad, y a los desarrolladores no frustrarse cuando no se logran resultados de una tarea, dado que es una posibilidad inherente a la naturaleza del proyecto.

Otro de los aprendizajes relevantes del proyecto fue la importancia de realizar una *demo* para el avance del proceso de desarrollo. Esto generó una práctica que luego se expandió al resto de los proyectos de la empresa.

El desarrollo del proyecto en sí involucró además el desarrollo de competencias técnicas y la inversión en capacitación y formación del equipo en los lenguajes de bases de datos MQL, SPARQL y en la estructura de DBpedia y Freebase, que antes no se utilizaban.

Se destacan además importantes vinculaciones que les permitieron incrementar sus capacidades, funcionar con soporte en el proceso de desarrollo y contar con desarrollos de terceros como insumo. La comunidad Python concentra el grueso de las vinculaciones tecnológicas más relevantes para el proyecto, y en segunda medida las relaciones con el ámbito universitario. La comunidad Python funcionó durante todo el proceso de desarrollo del proyecto anterior probando los distintos avances, y fue precisamente un miembro de esta comunidad quien identificó la posibilidad de desarrollar independientemente la pieza de SOSE que dio origen a Quepy. Además, en el proceso de desarrollo de Quepy se utilizaron librerías de código abierto realizadas en *Python* por terceros (por ejemplo, NLTK[53]), y al incorporarlas se hicieron consultas acerca de cómo implementarlas y se recibió soporte en ese sentido. Luego la interacción continuó, difundiendo lo hecho al momento de liberar el producto. La vinculación con la universidad se dio directamente a través de quienes fueron desarrolladores del proyecto, Laura A. y Rafael C., profesores en la Licenciatura en Computación de la Facultad de Matemática, Astronomía y Física de la Universidad Nacional de Córdoba. Ellos incorporaron Quepy como herramienta de formación en la asignatura que dictan, y ésta ha sido incorporada en la realización de distintas tesis.

- *Resultados de innovación y medidas de performance.*

Un parámetro de *performance* del proyecto es la medida en que satisfizo la necesidad estratégica que le dio origen. En este sentido, Quepy cumplió el objetivo de posicionar y exponer a la empresa, que era su principal meta, tanto a través de la liberación del producto como a través de la generación de presentaciones en conferencias (no académicas) y premios en concursos. La aplicación ganó en 2012 una mención especial otorgada en el "V Encuentro Nacional de Gobierno Electrónico" como parte del "Concurso de Datos Abiertos del Gobierno de Uruguay", en la categoría "ideas abiertas" de la presentación *Búsqueda por lenguaje natural en catálogo de datos abiertos*[54]. En 2014 la empresa aprovechó el desarrollo de Quepy para participar del evento *PyData Silicon Valley 2014* realizado en las oficinas de Facebook, donde presentaron la exposición "*Querying your Database in Natural Language"*[55].

La potencialidad que generan este tipo de proyectos terminados para participar de conferencias es uno de los principales resultados de innovación comercial para la empresa.

[52] Entrevista a Elías A., *Project Manager* de Quepy e impulsor de la idea del proyecto 18/06/2014.
[53] Más información en: http://www.nltk.org/
[54] http://www.agesic.gub.uy/innovaportal/v/2436/1/agesic/tengo_una_idea:_entrega_de_reconocimientos_de_dateideauy_.html.
[55] http://pydata.org/sv2014/abstracts/#197.

Las presentaciones en conferencias y la liberación del producto son medios para difundir las capacidades de desarrollo con que cuenta la empresa, dado que la mecánica interna de la aplicación queda expuesta. La participación en este tipo de eventos constituye también una oportunidad para el desarrollo de *networking* y acumulación de capital social con productores en sus áreas de experticia.

De este modo, se observa que el grueso de los resultados en materia de innovación del proyecto se encuentra en las dimensiones organizacional y comercial (vías de posicionamiento de mercado).

El grado de novedad técnica de Quepy no radica en su aporte a la ciencia, lo novedoso es la aplicación de un conocimiento existente a nuevos usos[56], su grado de simplicidad y su carácter *open source*[57].

Machinalis no obtiene en la actualidad ingresos, ni por la venta de la aplicación (que es de libre acceso) ni por servicios de soporte para esta sino que los beneficios materiales del proyecto son de tipo indirecto.

4. ANÁLISIS COMPARATIVO

El análisis comparativo de los dos tipos de proyectos permite hallar similitudes relevantes entre los proyectos para clientes y los proyectos internos que caracterizan a una empresa innovadora del sector como Machinalis. De esta manera, como sistematiza el cuadro 4, pueden identificarse una serie de elementos coincidentes entre ambos tipos de proyectos en lo referido a la motivación en el origen de los mismos, la organización del trabajo, los procesos de desarrollos de competencias técnica y organizacionales, la importancia de la conectividad y el carácter de las innovaciones introducidas, que sugieren la existencia de un patrón común de innovación al interior de la firma basado en la posibilidad de introducir innovaciones organizacionales que permitan el aprovechamiento de las capacidades y motivaciones del personal y la alta conectividad, con el objeto de generar innovaciones técnicas que permitan la supervivencia y crecimiento de la firma.

Por su parte, los principales rasgos distintivos entre estos tipos de proyectos se encuentran en el origen, naturaleza y finalidad de cada uno (el proyecto para clientes resulta de un proceso típico de negocios, en tanto que el proyecto interno resulta de un proyecto anterior y una estrategia específica de innovación de la empresa); en la importancia relativa de distintas fuentes de conectividad (en el caso del proyecto interno aparecen con mucha más fuerza las vinculaciones con la comunidad *open source*, mientras que en el caso del proyecto comercial las relaciones directas con el cliente tienen una relevancia primordial seguidas por las vinculaciones con instituciones de CyT y la comunidad *open source*); y en algunas medidas de sus resultados de innovación. Específicamente, existen en el caso del proyecto interno algunas medidas particulares de *performance* innovativa en materia de comercialización y *marketing* de la empresa, que la capacidad del proyecto para generar oportunidades de *networking* de negocios a través de las presentaciones en conferencias, la exposición pública mediante la liberación del producto y la obtención de premios en

[56] Rafael C., impulsor de la idea del proyecto y desarrollador en el mismo, sostiene: *"la idea de resolver ese problema con esa técnica, es de los '60"* (entrevista realizada el 28/05/2014), pero la técnica hoy se denomina distinto y el problema ha sido adaptado a los datos disponibles actualmente, esto es lo innovador.

[57] Respecto al producto completo, hay productos similares que tienen 5 años de antigüedad, como Wolfram Alpha, Siri (de Apple) y Google Now. Sin embargo, estos productos son de código cerrado y Quepy se ubica en este nicho pero, por su parte, es la única pieza aplicada, no puramente académica, que es *open source* y está disponible libremente para resolver los mismos problemas, aunque un poco más básico.

concursos. Ello sugiere algunas medidas especiales de resultados de innovación para ser tenidas en cuenta en los instrumentos de medición estadística.

CUADRO 4: ANÁLISIS COMPARATIVO DE DIMENSIONES RELATIVAS A INNOVACIÓN (*PROFLOW* – QUEPY). PRINCIPALES SIMILITUDES

Dimensión	Características
Origen de los proyectos	- Carácter tecnológicamente desafiante. - Alta relevancia del factor motivacional. - Objetivo de posicionamiento de mercado.
Organización del proceso de trabajo	- Alta especificación y plasticidad. - Adaptaciones de Metodologías Ágiles *Scrum* según i) naturaleza del proyecto, ii) características y cultura organizacional de la empresa, iii) necesidades y características de los integrantes del equipo. - Superación de dificultades en la gestión de los tiempos de trabajo.
Competencias técnicas	- Alta calificación en áreas de experticia: formación académica muy por encima de niveles promedio del sector y en segmentos *top knowledge* de la producción de *software*.
Competencias "blandas"	- Desarrollo de mecanismos efectivos de comunicación entre los integrantes del proyecto y con el cliente: habilidades sociales como capacidad de comprender y reelaborar necesidades de terceros, comunicar ideas y resultados, trabajar en equipo. - Soporte de la actividad productiva y el despliegue de las competencias técnicas.
Conectividad	Alta relevancia como fuente de conocimientos y desarrollo de competencias (clientes, comunidad *Open Source*).
Modalidades de aprendizaje predominantes	- *Learning by failing* al enfrentarse a dificultades. - *Learning by interacting* al participar de comunidades *open source* y vincularse con el cliente.
Resultados de innovación	Concentrados en materia organizacional y, en menor medida, en comercialización y posicionamiento de mercado.
Carácter de las innovaciones técnicas introducidas	- Novedad en la la aplicación de conocimientos científico-tecnológicos existentes a un problema o caso específico. - Grado de innovación técnica de los proyectos materializado en aspectos no necesariamente visibles, mensurables o cuantificables (innovaciones y mejoras en partes del producto y en desarrollos que se liberan aportando al corrimiento de la frontera de posibilidades de las herramientas de desarrollo utilizadas). - Innovación de producto: producto comercial antes inexistente, producto antes inexistente en *software* abierto.

Fuente: Elaboración propia

5. COMENTARIOS FINALES

A lo largo de este trabajo hemos analizado en términos comparativos el desarrollo de dos proyectos tecnológicos en una empresa innovadora del sector de SSI de Córdoba, Argentina. En particular, examinamos el origen de dichos proyectos, las particularidades que asume la organización del trabajo, el papel de las competencias técnicas y organizacionales y las vinculaciones, desde una perspectiva del desarrollo de capacidades y de la dinámica de la innovación en la firma.

En este espacio nos interesa entonces resaltar una serie de aspectos relativos a la dinámica de la innovación al interior de la firma, que resultan significativos en relación con los aportes de la literatura existente.

En primera instancia, nos interesa destacar aquí las evidencias que presenta el caso en relación con la manera en que las actividades innovativas forman parte del propio proceso productivo de las actividades de desarrollo de *software* y provisión de servicios asociados, lo que podría caracterizar una especificidad de la innovación en la producción de *software* distintiva respecto de las manufactureras.

En segundo término, se observa que efectivamente las conductas creativas durante el desarrollo de los proyectos habilitaron la emergencia de innovaciones, principalmente organizacionales, que permitieron, por un lado, superar las dificultades que las originaron y, por otro lado, ampliar el alcance de las metas inicialmente planteadas en términos de innovación. Esto es, ante las dificultades, no se optó por conductas predominantemente adaptativas que implicarían el abandono o minimización de los objetivos de los proyectos originalmente impulsados sino, en términos generales, por la implementación de estrategias de continuidad que dieron lugar a nuevas iniciativas de relevancia para la empresa en términos de innovación.

En tercer lugar, los aprendizajes organizacionales derivados de los proyectos permitieron a la empresa generar conocimientos que se materializaron en innovaciones que acabaron modificando prácticas internas al nivel de la empresa, extendiéndose a otros proyectos. En este caso, dado que la producción es a medida y no vuelve a producirse el mismo *software* (en serie, por ejemplo), la aplicación del nuevo conocimiento generado (la innovación organizacional) se materializa en la producción de algún otro *software* o proceso que excede al originario. Esta cuestión no deja de llamar la atención, como posible especificidad de los procesos de innovación en la actividad de *software* en relación a los sectores manufactureros donde tradicionalmente las nuevas prácticas organizacionales que surgen en el proceso productivo de un bien por lo general se incorporan a la producción del mismo bien más allá de que la innovación organizacional pueda o no extenderse a otra gama de productos de la empresa.

Por último, merecen una reflexión aparte las particularidades de la política de propiedad intelectual de la firma. Ello moviliza interrogantes en torno a las condiciones para la innovación que involucran las herramientas y comunidades *open source*. Las prácticas vinculadas al *software* abierto entran en tensión permanente con los derechos de propiedad privada y la necesidad de generación de ingresos dentro de una misma unidad productiva. Ello tanto en términos de las innovaciones técnicas que habilita esta forma de producir, como en términos de los cambios organizacionales que presupone e impulsa, que pueden constituir fuentes novedosas de innovación en un sector con alto potencial de desarrollo económico atravesado por lógicas de producción diversas aunque no necesariamente contradictorias. Dicha convivencia pone en tela de juicio las aseveraciones típicas de la literatura que relaciona la innovación radical con férreos derechos de propiedad intelectual.

6. BIBLIOGRAFÍA

Ancori, B., Bureth, A. y Cohendet, P. (2000): "The economics of knowledge: the debate about codification and tacit knowledge", *Industrial and Corporate Change*, 9(2), pp. 255-87.

Antonelli, C. (2008): *Localised technological change: towards the economics of complexity*, Routledge,

Arrow, K. J. (1962): "The economic implications of learning by doing", *The review of economic studies*, pp. 155-73.

Cowan, R., David, P. y Foray, D. (2000): "The explicit economics of knowledge codification and taciteness", *Industrial and Corporate Change*, 9(2), pp. 211-53.

Denzin, N. K. y Lincoln, Y. S. (eds.) (2005): *The SAGE handbook of qualitative research*. Sage.

Doloreux, D. y Shearmur, R. (2013): "Innovation Strategies: Are Knowledge-Intensive Business Services Just Another Source of Information?", *Industry and innovation*, 20(8), pp. 719-38.

Drejer, I. (2004): "Identifying innovation in surveys of services: a Schumpeterian perspective", *Research Policy*, 33(3), pp. 551-62.

Eisenhardt, K. M. (1989): "Building theories from case study research", *Academy of management review*, 14(4), 5 pp. 32-50.

Ernst, D. y Lundvall, B. Å. (2004): "Information technology in the learning economy: challenges for developing countries", en E. Reinert, *Globalization, Economic Development and Inequality*, GB: Edward Elgar Publishing.

Foray, D. y Lundvall, B. Å. (1998): "The knowledge-based economy: from the economics of knowledge to the learning economy", *The economic impact of knowledge*, pp. 115-21.

Gallouj, F. y Savona, M. (2009): "Innovation in services: a review of the debate and a research agenda", *Journal of Evolutionary Economics*, 19(2), pp. 149-72.

Gallouj, F. y Weinstein, O. (1997): "Innovation in services", *Research Policy*, 26(4), pp. 537-56.

He, Z.-L. y Wong, P.-K. (2009): "Knowledge interaction with manufacturing clients and innovation of knowledge-intensive business services firms", *Innovation*, 11(3), pp. 264-78.

Hertog, P. D. (2000): "Knowledge-intensive business services as co-producers of innovation", *International Journal of Innovation Management*, 4(04), pp. 491-528.

Lundvall, B. Å. (1996): "The social dimension of the learning economy", Druid Working Paper nº. 96-1.

Lundvall, B. Å. (1988): "Innovation as an interactive process: from user-producer interaction to the national system of innovation", en G. Dosi, C. Freeman, R. R. Nelson, G. Silverberg y L. Soete, *Technical change and economic theory.*

Lundvall, B. Å. ed. (1992): *National Systems of Innovation: towards a theory of innovation and interactive learning*, London: Printer Ed.

Lundvall, B. Å. y Johnson, B. (1994): "The learning economy", *Journal of industry studies*, 1(2), 2 pp. 3-42.

Maidique, M. A. y Zirger, B. J. (1985): "The new product learning cycle", *Research Policy*, 14(6), 2 pp. 99-313.

Miles, F., Wensley, A., Alba, M. y Álvarez-Coque, J. M. G. (2011): "How much does KIBS contribute to the generation and diffusion of innovation?", *Service Business*, 5(3), pp. 195-212.

Miles, I. (2005): "Knowledge intensive business services: prospects and policies", *Foresight*, 7(6), pp. 39-63.

Miozzo, M. y Grimshaw, D. (2005): "Modularity and innovation in knowledge-intensive business services: IT outsourcing in Germany and the UK", *Research Policy*, 34(9), pp. 1419-39.

Muller, E. y Doloreux, D. (2009): "What we should know about knowledge-intensive business services (KIBS)", *Technology in Society*, 31(1), pp. 64-72.

Neergaard, H. (2007): "Sampling in entrepreneurial settings", en H. Neergaard y J. P. Ulhøi, *Handbook of qualitative research methods in entrepreneurship.*

Neergaard, H. y Ulhøi, J. P. eds. (2007): *Handbook of qualitative research methods in entrepreneurship*, Edward Elgar Publishing.

Nelson, R. R. (1982): *An evolutionary theory of economic change*, Harvard University Press,

Niosi, J., Athreye, S. y Tschang, T. (2012): "The global computer *software* sector", *Economic Development As a Learning Process: Variation Across Sectoral Systems*.

Nonaka, I. (1995): *The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation: How Japanese Companies Create the Dynamics of Innovation*, Oxford university press,

Rosenberg, N. (1982): *Inside the black box: technology and economics*, Cambridge University Press,

Schumpeter, J. A. (1947): "The creative response in economic history", *The journal of economic history*, 7(2), pp. 149-59.

Sutherland, J. y Schwaber, K. (2012): "The Scrum Papers: Nut, Bolts, and Origins of an Agile Framework ", Cambridge, MA: Scrum Inc.

Takeuchi, H. y Nonaka, I. (1986): "The new new product development game", *Harvard business review*, 64(1), pp. 137-46.

Yin, R. K. (2009): *Case study research: Design and methods*, Sage.